

Chapter Eight: Contents

(AS-6 – 15 October 2001 – LA-UR 01-5716 – Portland Study Reports)

1. SCRIPTS	1
1.1 FIXWALKS.CSH.....	1
1.2 FIXLONGWALKS.CSH.....	2
1.3 FIXLONGWALKS.PL.....	8

Chapter Eight—AS-6

NOTE: Long code lines that do not fit completely on one line of this document are shown in italics and continued on to the next line.

1. SCRIPTS

1.1 fixwalks.csh

```
#!/usr/local/bin/tcsh -f

mkdir HH
cd HH/
/home/projects/transims/config/integration/bin/MakeHouseholdFile ../fixwalks.cfg.base 14
cd ../

../../scripts/FixLongWalks.csh > & ! FixLongWalks.log &
```

1.2 FixLongWalks.csh

```
#!/usr/local/bin/tcsh -f

#####

# This script goes through an iteration database and finds long walks,
# creates an activity regenerator feedback file using FixongWalks.pl,
# (re) runs the activity regenerator, recreates the iteration
# database, and repeats the whole process until the number of entries
# in the feedback file (equal to the number of trips with long walks)
# is smaller than a given threshold (MAXBAD). In each iteration, the
# location choice model's location coefficient multiplier is
# multiplied by MULTMULT and the model's time exponent is multiplied
# by EXPMULT.

#####

set MULTMULT = 1.05
set EXPMULT = 1.03
set MAXBAD = 200

set NUMPROCS = 4
set BINDIR = $INTEGRATION/bin
set SCRDIR = ~jpsmith/TRANSIMS/Feedback/allstr/scripts
set RUNDIR = $PWD
set OUTDIR = $RUNDIR
set LOGDIR = $OUTDIR/logs

set FDBK = $OUTDIR/walks.act.fdbk
set ITDB = $OUTDIR/itdb.000.it
set ITDBPREFIX = $OUTDIR/itdb
set CONFIG = $RUNDIR/fixwalks.cfg

#####

cd $RUNDIR
cp fixwalks.cfg.base fixwalks.cfg
```

```
#####

set ITER = 1
set MULT = $MULTMULT
set EXP = 0.5
set BAD = `echo $MAXBAD +1 | bc`

# re-index activities, since they are sometimes moved from other locations
eval ` $TRANSIMS_HOME/bin/SetEnv csh $CONFIG | grep SEL_ACTIVITY_FILE `
rm $SEL_ACTIVITY_FILE.idx
$BINDIR/IndexActivityFile $SEL_ACTIVITY_FILE

while( $BAD > $MAXBAD )
  set test = `echo "scale=6; 0.08*$MULT*e($EXP*1(16500))" | bc -l`
  echo "Iteration $ITER, MULT=$MULT, EXP=$EXP, MaxArg=$test, lastBAD=$BAD>$MAXBAD"

  # prep for this iteration
  if( ! -e $LOGDIR ) mkdir $LOGDIR
  set TEST = "new"

  # execute this iteration:
  # run the collator
  echo "Collator";
  echo "  start:" `date`
  @ I = 0
  while( $I < $NUMPROCS )
    $BINDIR/Collator $CONFIG $I >&! $LOGDIR/Collator.$I.log &
    @ I++
  end
  # wait for all Collators to finish
  sleep 30
  while( "$TEST" != "" )
    set TEST = `ps -u jpsmith | grep Collat`
    sleep 10
  end
  echo "  finish:" `date`
  sleep 10
  # merge into single file, remove old ones
  head -2 $ITDBPREFIX.tAA.000.it >! $ITDB
  foreach I ( `ls $ITDBPREFIX.t???.000.it` )
```

```
    gawk 'NR>2' $I >> $ITDB
end
rm $ITDBPREFIX.t???.000.it $ITDBPREFIX.t???.idx

echo "FixLongWalks"
echo " " `time $SCRDIR/FixLongWalks.pl $ITDB $FDBK $MULT >&! $LOGDIR/FixLongWalks.log`
echo "ActivityRegenerator"
echo " " `time $BINDIR/ActivityRegenerator $CONFIG`

# check for errors
echo `grep ERR $LOGDIR/* | wc -l | gawk '{print $1 " total errors."}'`

# prep next iteration (may not do - test at top of loop)
echo "prep next"
set MULT = `echo "$MULTMULT * $MULT" | bc`
set EXP = `echo "$EXPMULT * $EXP" | bc`
set BAD = `wc -l $FDBK | gawk '{print $1}'`

echo "" >> $CONFIG; echo "# Fix-Walks Iteration `echo "$ITER+1" | bc`" >> $CONFIG
echo "SEL_ACTIVITY_FILE $OUTDIR/iter_$ITER/activities.partial" >> $CONFIG
echo "ACTIVITY_FILE $OUTDIR/iter_$ITER/activities.partial" >> $CONFIG
echo "ACT_LOCATION_CHOICE_EXPONENT $EXP" >> $CONFIG

# archive results of this iteration
echo "archive"
mkdir $OUTDIR/iter_$ITER
rm $OUTDIR/*.idx
mv $ITDB $OUTDIR/iter_$ITER/
mv $FDBK $OUTDIR/iter_$ITER/
mv $OUTDIR/activities.partial $OUTDIR/iter_$ITER/
mv $OUTDIR/activity.problems $OUTDIR/iter_$ITER/

echo "Index last Activity File"
echo " " `time $BINDIR/IndexActivityFile $OUTDIR/iter_$ITER/activities.partial >&!
$LOGDIR/IndexActivityFile.log`

mv $LOGDIR $OUTDIR/iter_$ITER/

@ ITER++
end
```

```
#####
# make what remains into mode 8

    echo "Iteration $ITER, using mode 8 for remainder, lastBAD=$BAD>$MAXBAD"

# prep for this iteration
if( ! -e $LOGDIR ) mkdir $LOGDIR
set TEST = "new"
echo "ACT_LOCATION_CHOICE_EXPONENT 0.5" >> $CONFIG # so no unexpected crashes

# execute this iteration:
# run the collator
echo "Collator";
echo "  start:" `date`
@ I = 0
while( $I < $NUMPROCS )
    $BINDIR/Collator $CONFIG $I >&! $LOGDIR/Collator.$I.log &
    @ I++
end
# wait for all Collators to finish
sleep 30
while( "$TEST" != "" )
    set TEST = `ps -u jpsmith | grep Collat`
    sleep 10
end
echo "  finish:" `date`
sleep 10
# merge into single file, remove old ones
head -2 $ITDBPREFIX.tAA.000.it >! $ITDB
foreach I ( `ls $ITDBPREFIX.t???.000.it` )
    gawk 'NR>2' $I >> $ITDB
end
rm $ITDBPREFIX.t???.000.it $ITDBPREFIX.t???.idx

echo "FixLongWalks"
echo "  " `time $SCRDIR/FixLongWalks.pl $ITDB $FDBK $MULT >&! $LOGDIR/FixLongWalks.log`

# convert all relocate commands into remode commands
mv $FDBK $FDBK.tmp
sed 's/LTR '$MULT'/MS 8/g' $FDBK.tmp > $FDBK
rm $FDBK.tmp
```

```
echo "ActivityRegenerator"
echo " " `time $BINDIR/ActivityRegenerator $CONFIG`

# check for errors
echo `grep ERR $LOGDIR/* | wc -l | gawk '{print $1 " total errors."}'`

# archive results of this iteration
echo "archive"
mkdir $OUTDIR/iter_$ITER
rm $OUTDIR/*.idx
mv $ITDB $OUTDIR/iter_$ITER/
mv $FDBK $OUTDIR/iter_$ITER/
mv $OUTDIR/activities.partial $OUTDIR/iter_$ITER/
mv $OUTDIR/activity.problems $OUTDIR/iter_$ITER/

echo "Index last Activity File"
echo " " `time $BINDIR/IndexActivityFile $OUTDIR/iter_$ITER/activities.partial >&! $LOGDIR/IndexActivityFile.log`

mv $LOGDIR $OUTDIR/iter_$ITER/

#####
# merge everything

if( ! -e $LOGDIR ) mkdir $LOGDIR

echo "Merge Activity Indices"
set PARTS = `ls -rt iter_*/activities.partial.hh.idx`
echo " " `time $BINDIR/MergeIndices AS-6.hh.idx
/home/jpsmith/TRANSIMS/Feedback/allstr/AS3.25/fixtransit/AS-5.hh.idx $PARTS >&! $LOGDIR/MergeIndices.log`

echo "Defragment Activities"
echo " " `time $BINDIR/IndexDefrag AS-6.hh.idx AS-6 >&! $LOGDIR/IndexDefrag.log`

echo "Make route file"
cat ../fixtransit/route.DF ../fixtransit/route.fix_PnR iter_*/walks.act.fdbk | gawk '{print $1}' | sort -n
| uniq > AS-4-5-6.HH.fdbk
```

```
#####  
#cp fixwalks.cfg $OUTDIR/  
  
exit
```

1.3 FixLongWalks.pl

```

#!/usr/local/bin/perl -w

#####

# This script goes through an iteration database and finds long walks,
# then creates an activity regenerator feedback file to increase the
# location coefficient multiplier to that given on the command line.

# It should be used iteratively with the activity regenerator: if
# after any iteration a traveler still has a long walk, that traveler
# should be relocated with an even higher multiplier.

#####

$delim = ",";
$heads = 2;

# get command line arguments: iteration database and feedback command filenames
local($itdb, $fdbk, $mult) = @ARGV;
print "Mode coefficient multiplier is $mult.\n";

# open input file
open(ITDB, "$itdb") || die "Failed to open $itdb for input ($OS_ERROR).";
# read header lines
for($i=0;$i<$heads;$i++) {
    $line = <ITDB>;
}
# find fields automatically using header
chomp $line;
@data = split($delim, $line);
$i=0;
$HHField = $Act2Field = $Type1Field = $Type2Field = $ModeField = $DistField = -1;
while($data[$i]) {
    if($data[$i] eq "HH") {
        $HHField = $i;
    }
}

```

```
    elif($data[$i] eq "END_ACT_ID") {
        $Act2Field = $i;
    }
    elif($data[$i] eq "START_ACT_TYPE") {
        $Type1Field = $i;
    }
    elif($data[$i] eq "END_ACT_TYPE") {
        $Type2Field = $i;
    }
    elif($data[$i] eq "END_MODE_PREF") {
        $ModeField = $i;
    }
    elif($data[$i] eq "EUCLID") {
        $DistField = $i;
    }
    $i++;
}
# check for mandatory fields
$Missing = 0;
if($HHField == -1) {
    print "Missing HH field in itdb.\n";
    $Missing++;
}
if($Act2Field == -1) {
    print "Missing END_ACT_ID field in itdb.\n";
    $Missing++;
}
if($Type1Field == -1) {
    print "Missing START_ACT_TYPE field in itdb.\n";
    $Missing++;
}
if($Type2Field == -1) {
    print "Missing END_ACT_TYPE field in itdb.\n";
    $Missing++;
}
if($ModeField == -1) {
    print "Missing END_MODE_PREF field in itdb.\n";
    $Missing++;
}
if($DistField == -1) {
    print "Missing EUCLID field in itdb.\n";
```

```

    $Missing++;
}
if($Missing > 0) {exit;}

#####

open(FDBK, ">$fdbk") || die "Failed to open $fdbk for output ($OS_ERROR).\n";

$count1 = 0;
$count2 = 0;
while ( <ITDB> ) {
    $line = $_; chomp $line;
    @data = split($delim, $line);

    if(($data[$ModeField] eq 1 && $data[$DistField] > 2500)
        || ($data[$ModeField] eq 7 && $data[$DistField] > 7500)) { # long walk or long bike
        if($data[$Type1Field] eq 7 || $data[$Type2Field] eq 7) { # school
            print FDBK "$data[$HHField] $data[$Act2Field] MS 9\n"; # walk/bike -> school bus
            ++$count1;
        }
        else {
            print FDBK "$data[$HHField] $data[$Act2Field] LTR $mult\n"; # re-locate, new multiplier
            ++$count2;
        }
    }
}

close ITDB;
close FDBK;

#####

print "Found $count1 long school trips and $count2 long other trips\n";

exit;

```